CLOCK TREE SYNTHESIS FOR A HIERARCHICALLY PARTITIONED IC LAYOUT

BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION

[0001] The present invention relates in general to computer-aided design tools for generating hierarchical integrated circuit (IC) layouts, and in particular to a method for synthesizing a clock tree for a hierarchically partitioned IC layout.

DESCRIPTION OF RELATED ART

Hierarchical IC Layouts

[0002] A "cell library" is a data base of models of various circuit components ("cells") an integrated circuit (IC) designer can incorporate into an IC design. The cell library includes behavioral models of each cell and also a physical model of each cell showing placement and routing (P&R) tools how the cell is to be laid out in an a semiconductor substrate when included in an IC.

[0003] An IC designer typically creates a "netlist" describing the IC by referencing the library cells forming it and indicating how the terminals of the cells are logically interconnected. After having created the netlist, and having used a circuit simulator and other tools to the verify the behavior of the circuit described by the netlist, an IC designer typically provides the netlist as input to an automated P&R tool which designs the physical layout of the IC on a semiconductor substrate. The P&R tool consults the cell library to determine how to lay out each cell described by the netlist and then determines where to place each cell in the substrate and how to form signal, power and ground paths for connecting the cells to one another.

[0004] The netlist that the designer produces usually describes an IC as a hierarchy of modules. For example as illustrated in FIG. 1, the netlist may consider an entire IC design to be a single top level (level 1) module M0. A set of three level 2 modules M1-M3 and some individual cells 10 not included in any of modules M1-M3 form module M0. Each module M1-M3 may in turn be formed by level 3 modules or cells. In the example of FIG 1, module M2 includes a set of four level 3 modules M4-M7 and some individual cells 11 not included in any of modules M4-M7. Modules M4-M7 reside at the lowest level (level 3) of the modular hierarchy because they are formed only by base level cells 12 which are not

organized into submodules. Although FIG. 1 illustrates a circuit design having only a few modules organized into a three hierarchical levels, a large IC design may include hundreds or thousands of modules organized into many more hierarchical levels.

[0005] An IC designer organizes a IC design as a hierarchy of modules to make it easier to conceptualize the IC. However since a typical P&R tool is interested only in finding a suitable position for each cell of the design, it ignores the modularity of the design and places each cell wherever it is convenient to do so and does not try to keep all of the cells of a given module within a separately identifiable area of a semiconductor substrate. Hence the cells forming various modules of a hierarchical design usually end up being intermingled to some extent in an IC layout.

Partitioned Layouts

[0006] A designer may want to produce a "partitioned" IC layout in which one or more selected modules are to be placed in separate partitions of a substrate. For example FIG. 2 illustrates a floor plan for a semiconductor substrate 16 upon which the IC of FIG. 1 may be formed. FIG. 1 illustrates how the IC layout of FIG. 1 might be partitioned. In this example, module M1 is placed within partition P1, modules M5 and M7 are placed within partition P2, modules M4 and M6 are placed in partitions P3 and P4, respectively. All modules and cells not included in any "base level" partition P1-P4 (in this example only module M3 and cells 10 and 11 of modules M0 and M2_) may be placed anywhere within a "top level" partition P0. For simplicity partitions P0- P4 are illustrated as being rectangular, but partitions may be of more complicated shapes. Given a pin assignment plan indicating the places along the boundaries of the various partition where signal paths are to cross between partitions, P&R tools can independently lay out each partition, provided that they are able to keep the layouts within the partition boundaries specified by the floor plan.

[0007] It can be advantageous to partition a design because a P&R tool can often separately layout several partitions of a design faster than it can lay out an unpartitioned design. Also when a designer places a selected module of a design in an identifiable area of a substrate so that its cells are not intermingled with cells of other modules, the designer may be able to later modify that particular module without greatly affecting the layout of other modules. However, as discussed below, problems associated with clock tree synthesis tend to increase the interdependence of

partition layouts, thereby complicating the layout process after one partition is changed.

Clock Tree Synthesis

[0008] Digital ICs typically implement synchronous logic circuits that are clocked by externally generated clock signals. For example FIG. 3 shows a simple example of a synchronous logic circuit within an IC 30. A latch 30 latches two IC input signals A and B onto inputs of a logic circuit 34 and an IC input signal C onto an input of a logic circuit 36. Logic circuit 34 carries out logic operations on signals A and B to produce an output signal D latched onto another input of logic circuit 36 by a latch 38. Logic circuit 36 processes signals C and D to form a signal E, and a latch 39 latches signal E onto an output terminal of IC 30. A CLOCK signal supplied as input to IC 30 clocks all of latches 30, 38 and 39. For the circuit to operate synchronously, it is necessary to route the CLOCK signal to each latch 36, 38 and 39 in such a way that edges of the CLOCK signal arrive substantially at the same time at all latches.

[0009] While FIG. 3 shows only three devices being clocked by the CLOCK signal, a typical IC can have hundreds or thousands of clocked devices ("syncs") such as latches, registers and flip-flops. After a P&R tool has generated an IC layout in which all of the cells (including all of the syncs) have been placed and routed, a "clock tree synthesis" tool designs a clock tree for routing CLOCK edges from the IC's clock input terminal concurrently to all syncs.

[0010] FIG. 4 is a schematic diagram illustrating a clock tree 40 for delivering a CLOCK signal from an IC input pin 42 to several syncs 44. FIG. 5 illustrates positions of syncs 44 within an IC substrate 46 after a P&R tool has generated a full-chip IC layout. A typical clock tree synthesis tool begins designing the clock tree of FIG. 4 by assigning each sync 44 into one of a set of "clusters" 48 as illustrated in FIG. 6. The clock tree will deliver the CLOCK signal to the syncs 44 of each cluster 48 through a separate one of signal buffers 50 (FIG. 4), and therefore each cluster 48 will include no more than the maximum number of syncs 44 a single buffer 50 can drive. To ensure that CLOCK signal edges travel from the output of each buffer 50 to all syncs 44 it drives with as nearly as possible the same delay, the clock tree synthesis tool groups only nearby syncs 44 into the same cluster 48. This ensures that the CLOCK signal path distance from each buffer 50 to the syncs it

drives will be substantially similar regardless of where the buffer is placed.

[0011] Having grouped all syncs 44 into clusters and having specified an approximate location for each buffer 50 near the cluster it drives, the synthesis tool next groups nearby buffers 50 into clusters. As illustrated in FIG. 4, the CTS tool has grouped twelve buffers 50 into four clusters 52. The CTS tool then provides a separate buffer 54 to drive each cluster 52 of buffers 50. After specifying an approximate location for each second level buffer 54 near the cluster of first level buffers 50 it is to drive, the CTS tool groups buffers 54 into clusters 56, each driven by a separate buffer 58. All buffers 58 are then grouped into a single cluster 60 to be driven by a single buffer 62. The result is a hierarchical clock tree 40 having four levels of buffers 50, 54, 58 and 62.

[0012] Having laid out the basic framework of clock tree 40, the CTS tool next must "balance" the clock tree to ensure that each CLOCK signal edge arrives at all syncs 44 at as nearly as possible the same time, thereby substantially zeroing the CLOCK signal skew between syncs 44. To do so the CTS tool may insert additional buffers 64 into various branches of tree 40, since adding a buffer 64 to a branch reduces the signal path delay through that branch. The CTS tool may also adjust the size of selected buffers in each clock signal path; large buffers switch faster than small buffers and therefore propagate CLOCK signal edges faster. A CTS tool may also make fine adjustments to path delays by adjusting the positions of the buffers along a path. By adjusting the number, size and positions of buffers within each branch of clock tree 40, a "zero skew" CTS tool can synchronize the arrival of CLOCK signal edges at a large number of syncs with a high degree of accuracy.

[0013] FIG. 7 illustrates the steps in a conventional process for producing an IC layout when a design has been partitioned. The first step (step 70) is to create a floor plan reserving a separate area of the substrate for each partition P0-P4 of the design and a pin assignment plan indicating points at which signal are to cross boundaries of each partition. P&R tools then separately lay out each partition P0-P4 (step 72). After the partition layouts are combined to form a full-chip layout (step 74), a CTS tool synthesizes a clock tree for the entire IC (step 76). As it synthesizes a clock tree, the CTS tool establishes only an approximate position for each buffer forming the clock tree and only approximates the routing of signal paths that will link the

buffers to one another and to the syncs so that it can make reasonably accurate estimates of signal path delays through the clock tree. After the CTS tool has established approximate positions of clock tree buffers and signal paths, the P&R tool must globally adjust the layout of the entire IC to position each buffer forming the clock tree and to detail the routing of the signal paths that interconnect the buffers and syncs (step 78). In doing so, the P&R tool may move some of the cells forming the IC to make room for the clock tree buffers and clock tree signal routing paths. Thus even though a P&R tool may be able to separately lay out each partition of a design and then assemble the partitions into a full-chip layout, it must reprocess the entire full-chip layout at step 78, including the layout of each partition, following clock tree synthesis in order to place and route the buffers and signal paths forming the clock tree.

[0014] Conventional CTS tools operate only after the partitions have been merged because they must have a global view of clock signal path delays throughout the entire IC in order to balance the clock tree. Therefore, should the designer make a change to any partition of an IC design after the layout process is complete, it necessary for the designer to adjust not only the layout of that partition, but to also adjust the layout of every other partition of the design to accommodate global changes that a CTS tool might make in a clock tree when any small part of a layout changes. Thus while IC designs can initially be divided into partitions that can be separately laid out, any subsequent change to any partition that affects a clock tree can make it necessary to adjust the layout of every other partition.

[0015] What is needed is a clock tree synthesis system that can partition the design of a clock tree along the same lines that an IC is partitioned for layout purposes so that a change to the layout of any one partition requiring a change in the portion of the clock tree residing within that partition would not affect portions of the clock tree residing within any other partition and would therefore not require a change to the layout of any other partition.

SUMMARY OF THE INVENTION

[0016] The invention relates to a method and apparatus for synthesizing a clock tree for a partitioned integrated circuit (IC) layout, wherein the layout includes a plurality of base level partitions and a top level partition, with each partition occupying

a separate area of a semiconductor substrate. Each base level partition includes several syncs to be clocked by edges of a clock signal applied to an entry node within the area occupied by the top level partition.

[0017] In accordance with the invention, a subtree is separately synthesized for each base level partition. The subtree for each base level partition includes a start point at a perimeter of the area occupied by that partition and a network of buffers and signal paths for conveying a clock signal edge from the start point to each sync included within that area. The subtree for each base level partition is balanced so that it delivers each clock signal edge arriving at its start point to each sync within the base level partition with a substantially similar delay. However the subtrees for the separate base level partitions are separately and independently balanced and will normally have substantially differing average clock signal path delays between their start points and the syncs they clock.

[0018] After the clock tree for each base level partition is synthesized, a top level portion of the clock tree is synthesized. The top level portion of the subtree resides within the substrate area containing the top level partition and conveys the clock signal from the IC clock signal entry point to the start point of each synthesized subtree. Clock signal paths within the top level portion of the subtree are adjusted to compensate for differences in the average path delays of the subtrees so that the clock tree conveys each clock signal edge arriving at the IC entry point to every sync with substantially the same delay.

[0019] Once the clock tree has been synthesized, it is possible to re-synthesize the top level portion of the clock tree and the subtree for any one base level partition to accommodate a change in the layout of that one base level partition. It is not necessary to modify the subtree for any other base level partition.

[0020] Accordingly it is an object of the invention to provide a system for synthesizing a clock tree for a partitioned IC design, wherein changes in the layout of any base level partition do not necessitate changes to portions of the clock tree serving syncs residing in any other base level partition.

[0021] The claims appended to this specification particularly point out and distinctly claim the subject matter of the invention. However those skilled in the art will best understand both the organization and method of operation of what the applicant(s) consider to be the best mode(s) of practicing the invention,

together with further advantages and objects of the invention, by reading the remaining portions of the specification in view of the accompanying drawing(s) wherein like reference characters refer to like elements.

BRIEF DESCRIPTION OF THE DRAWINGS

[0022] FIG. 1 is a block diagram illustrating a prior art hierarchical integrated circuit (IC) design;

[0023] FIG. 2 is a prior art floor plan for the layout of the IC design of FIG. 1;

[0024] FIG. 3 is a block diagram illustrating a prior art synchronous circuit;

[0025] FIG. 4 is schematic diagram of a prior art integrated circuit clock tree;

[0026] FIG. 5 is a simplified plan view of a semiconductor substrate showing positions the syncs of FIG. 4 occupy within an IC layout;

[0027] FIG. 6 illustrates a manner in which a prior art clock tree synthesis tool might organize the syncs of FIG. 5 into clusters;

[0028] FIG. 7 is a flow chart illustrating a prior art IC layout and clock tree synthesis process;

[0028] FIG. 8 is a flow chart illustrating an IC layout and clock tree synthesis process in accordance with the present invention;

[0030] FIG. 9 is a floor plan of an IC layout upon which has been superimposed a symbolic depiction of a clock tree synthesized in accordance with the invention; and

[0031] FIG. 10 depicts a clock tree of FIG. 9 in schematic diagram form.

DESCRIPTION OF THE PREFERRED EMBODIMENT(S)

[0032] The present invention is directed to a method and apparatus for synthesizing a clock tree for an IC design that has been divided into partitions for separate placement and routing. The invention is suitably implemented in the form of a computer program stored on computer-readable media such as, for example, a compact disk so that the program may be read and executed by a conventional computer. This specification describes an exemplary embodiment and application of the invention considered by the applicants to be the best mode of practicing the invention.

[0033] An IC designer typically creates a netlist describing an IC as a hierarchy of modules. For example as illustrated in FIG. 1, the netlist may consider an entire IC design to be a single top level module M0 formed by a set of three lower level modules M1-M3 along with some individual cells 10 not included in any of modules M1-M3. Each module M1-M3 is in turn formed by lower level modules and cells. For example module M2 is formed by a set of four lower level modules M4-M7 and some individual cells 12 not included in any of modules M4-M7. In this example modules M4-M7 reside at the lowest level of the modular hierarchy because they are formed only by base level cells 14 which are not organized into submodules. Although FIG. 1 illustrates a circuit design having only a few modules organized into a few hierarchical levels, a large IC design may include a large number of modules organized into many hierarchical levels.

[0034] A designer sometimes divides an IC design into partitions so that a placement and routing (P&R) tool can separately lay out each partition. For example FIG. 2 illustrates a floor plan a designer might create when partitioning the hierarchical circuit of FIG. 1. Here the designer has specified that various modules are to be placed in particular rectangular partitions of an IC substrate 16. All cells forming modules M1, M4, and M6 are to be placed within partitions P1, P3, and P4, respectively. All cells forming modules M5 and M7 are to be placed within partition P2, though cells of the two modules may be intermingled with one another within that partition. All other modules and cells not assigned to any of the "base level" partitions may be placed anywhere within a "top level" partition P0 which includes all areas of substrate 16 not reserved for the base level partitions P1-P4. Thus module M3 and the individual cells 10 and 11 of modules M0 and M2 not included in any base level partition may be placed anywhere within top level partition P0.

[0035] In addition to a floor plan allocating substrate space to each partition, the designer provides a "pin assignment" plan indicating points along the boundaries of each partition at which signal paths conveying the input and output signals of modules included within each partition are to cross. P&R tools can then independently lay out each partition provided that they are able to conform the partition layouts to the floor and pin assignment plans. Such partitioning of the design can often speed up the layout process because it is usually faster for a P&R tool to lay

out several partitions of an IC than to lay out the entire IC at once.

[0036] As discussed above, an IC employing synchronous logic may include thousands of devices ("syncs") such as registers, latches and flip-flops that are clocked by the same clock signal (CLOCK) supplied as input to the IC. A clock tree synthesis (CTS) tool designs a clock tree as a network of buffers that delivers each edge of the CLOCK signal to all syncs with as little as possible difference (skew) between arrival times at the various syncs.

[0037] FIG. 7 illustrates a conventional approach to generating a layout for a partitioned IC design. After creating floor and pin assignment plans establishing the boundaries of each partition and points of interconnection between the partitions (step 70), all partitions are separately laid out in accordance with those plans (step 72). The partition layouts are then assembled into a full-chip layout (step 74) and a clock tree synthesis tool then synthesizes a full-chip clock tree for the entire IC (step 76). When synthesizing a clock tree, the CTS tool specifies only approximate positions of the buffers and the signal paths interconnecting them for purposes of estimating signal path delays within the clock tree. Thus after the CTS tool synthesizes the clock tree, the P&R tool must adjust the layout of every partition to place and route the buffers forming clock tree (step 78).

[0038] After completing the layout of an IC, a designer might like to be able to modify the layout of any one partition without affecting the layouts of all of the other partitions. However a conventional clock tree synthesis (CTS) tool cannot modify only the portion of the clock tree serving the changed partition; it has to re-synthesize the clock tree for the entire IC. Thereafter the designer would have to use a P&R tool to adjust the layout of every partition to accommodate the new clock tree. The present invention relates to a method for synthesizing a clock tree in a way that allows a portion of a clock tree serving modules of one partition to be changed without affecting the layout of any other partition except the top level partition.

[0039] FIG. 8 illustrates a partitioned IC layout and clock tree process in accordance with the invention. After floor and pin plans are created (step 80), and after a P&R tool separately lays out each partition (step 81) a CTS tool in accordance with the invention separately synthesizes the portion (subtree) of the clock tree that is to serve each partition (step 82).

[0040] FIG. 9 depicts the floor plan of FIG. 2 and uses large triangle to symbolically represent the subtrees T1-T4 synthesized for the four base partitions P1-P4 and the subtree T0 synthesized for the top level partition P0. The CTS tool may use any conventional "zero skew" algorithm to synthesize subtrees T0-T4 at step 82 in a way that minimizes the difference in clock signal path delay between each sync connected to the subtree and the starting point S0-S4 at which the clock signal enters the correspond subtree T0-T4. Conventional zero skew algorithms are capable of balancing each subtree T0-T4 To that the path delay from its start point S0-S4 to any one of the syncs served by subtree T0 varies little from the average path delay D0-D4 to all syncs served by that subtree. Since each subtree T0-T4 is independently synthesized, average path delays D0-D4 for the five subtrees T0-T4 may vary substantially from one another. For example when D3 is substantially greater than D4, it will take substantially longer for a CLOCK signal to travel from point S3 to any sync served by tree T3 than for the CLOCK signal to travel form point S4 to any sync served by tree T4.

[0041] After separately synthesizing each subtree at step 82, a P&R tool separately modifies the layout of each base level partition P1-P4 to fix the positions of buffers including in the partition's subtree T1-T4 and to route the signal paths linking the buffers to the subtree start point S1-S4 and to the syncs served by the subtree (step 83). The CTS tool next synthesizes a "top level" of the clock tree which links all of subtrees T0-T4 to an "entry" node N5 at which the CLOCK signal is to enter the IC from an external source. As illustrated in FIG. 9, clock tree top level 18 includes a set of buffered signal paths 18 formed in top level partition P0.

[0042] FIG. 10 illustrates in schematic diagram form the clock tree the CTS tool generates for the IC layout of FIG. 9. The CTS tool synthesizes the top level of the clock tree by designing paths 18 to sequentially merge all of the subtrees T0-T4 into the main clock tree, starting with any selected base level subtree T1-T4 and ending with the top level subtree T0.

[0043] Referring to FIGs. 9 and 10, the CTS tool first merges subtrees T4 and T3 by linking their starting points through buffered paths to a common node N1. The average CLOCK signal path delay D4 from subtree start point S4 to each sync served by subtree T4 may differ substantially form the average CLOCK signal path delay D3 from subtree start point S3 to each sync served by subtree

T3. Therefore the CTS tool designs the paths linking start points S4 and S3 to node N1 so that the average path delay from node N1 to any sync served subtrees T4 substantially matches the average path delay from node N1 to any sync served by subtree T3. When necessary it reduces the path delay between node N1 and either of points S4 or S3 by increasing the number and/or size of buffers 20 in the path. The CTS also adjusts the path delays by adjusting the position of node N1 relative to points S4 and S3. For example to increase the delay between node N1 and point S4 and decrease the delay between node N1 and point S3, the CTS tool can position point N1 closer to point S3 and farther away from point S4.

[0044] After having linked subtrees T4 and T3 to node N1, the CTS tool next synthesizes the portion of the clock tree linking node N1 and the start point S2 of subtree T2 to a common node N2. The CTS tool adjusts the size and number of buffers 20 in the path between nodes N2 and N1 and the path between node N2 and point S1 and the position of node N2 so that the average delay from node N2 to all syncs served by subtree T2 as nearly as possible matches the average delay from node N2 to all syncs served by subtrees T3 and T4.

[0045] The CTS tool then synthesizes buffered signal paths linking node N2 and the start point S1 of subtree T1 to a common node N3 and paths linking node N3 and the start point of subtree T0 to a common node N4. As it merges each subtree into the main clock tree, the CTS tool designs the interconnecting paths to substantially match the average clock signal path delay to the syncs served by the subtree being merged to the average clock signal path delay to syncs served by all previously merged subtrees. The CTS tool completes the synthesis of the top level of the clock tree by providing a buffered path linking node N4 to the node N5 at which the CLOCK signal arrives on substrate 16 via a bond wire or other type of external connection.

[0046] After the CTS tool has synthesized the top levels of the clock tree at step 84 (FIG. 8), a P&R tool adjusts the layout of top level partition P0 to detail the placement and routing of the top levels of the clock tree (step 85). The layouts of all partitions may then be assembled to produce a full-chip layout (step 86).

[0047] Note that in the prior art process illustrated in FIG. 7 partition layouts are assembled into a full-chip layout at step 74 before clock tree synthesis begins at step 76. In the layout process of the present invention, depicted in FIG. 8, clock tree

synthesis is separately carried out on each base and top level partition at steps 82 and 84 before the partitions are assembled into a full-chip layout. Thus the CTS tool is able to independently synthesize subtrees for all partitions P0-P4.

[0048] One advantage to the new process illustrated in FIG. 8 over the prior art process of FIG. 7 becomes apparent when the designer wants to change the layout of any partition of an IC design for which placement and routing has already been completed. As discussed above, when a conventional full-chip CTS tool is used to synthesize the clock tree for the entire IC, a change to the layout of any partition P0-P4 requires the CTS to re-synthesize the entire clock tree, thereby requiring the designer to adjust the layout of every partition. When the clock tree has been synthesized in accordance with the invention, then it is possible to change to the layout of any partition P0-P4 without requiring a change to the clock tree subtree serving any other partition. For example if the designer were to modify the layout of partition P3 of FIG. 9, it would be necessary for the CTS tool to re-synthesize only the subtree T3 for that partition and (if the path delay for the revised subtree T3 changes) to the top level portions of the clock tree formed in partition P0. However the CTS tool would not have to make any changes to the subtrees T0-T2 or T4 serving any other base level partition. Thus a change in the layout of partition P3 of FIG. 9 typically requires only relatively small changes in the layout of top level partition P0 as necessary to accommodate any needed adjustments to the top level the clock tree. Layouts of all other base level partitions remain unchanged.

[0049] In some applications it will not be necessary to synthesize a subtree for every partition as illustrated in FIGS. 9 and 10 because some partitions may not include any syncs. Thus for example in an application where all modules and cells are assigned to base level partitions, it would not be necessary to synthesize a subtree for the top level partition. In such case the top level partition would include buffer signal paths only for routing input/output signals between the various base level partitions and for routing the CLOCK signal from its entry point to those base level partitions that do include syncs.

[0050] The forgoing specification and the drawings depict a best mode of practicing the invention, and elements or steps of the depicted best mode exemplify the elements or steps of the invention as recited in the appended claims. However the appended claims are intended to apply to any mode of practicing the invention

comprising the combination of elements or steps as described in any one of the claims, including elements or steps that are functional equivalents of the example elements or steps depicted in the specification and drawings. Accordingly should any appended claim describe an element or step only in terms of its function, then it is intended that the claim's description of the element be interpreted as reading on any element or step having the described function, regardless of any structural limitations associated with any example of such element or step depicted in this specification or in the drawings.